Análisis y reporte del modelo.

Fabián González Vera A01367585

1. Introducción

Se desarrollo una implementación de una red neuronal convolucional utilizando el framework de TensorFlow. El fin de esta implementación es determinar la especie a la que pertenece un ave dada una imagen.

2. Dataset

El dataset[1] proviene del repositorio Caltech DATA del Instituto Tecnológico de California. El dataset contiene imágenes de 200 especies de aves y 11,788 imágenes además de anotaciones para cada imagen que contienen ubicaciones de partes, atributos binarios y bounding boxes.

El dataset fue reducido a 69 clases para reducir la dificultad del problema, esto fue a través del descarte de clases, como criterio para descarte se decidió comparar las diferentes subespecies y escoger la que tuviera la apariencia más única, la razón por la cual se opto por este criterio es que hay varias familias de aves y estas comparten muchas características, un ejemplo de esto son los gorriones de las cuales hay 7 subespecies en el dataset y estas poseen una apariencia bastante similar.

El dataset se dividido en 80% para entrenamiento y 20% para validación. Además, se realizó *data augmentation* al dataset para poder tener mas datos debido a que las clases tienen diferente cantidad de imágenes.

3. Modelo

Se desarrollaron varios modelos de redes convolucionales. Para la primera iteración se desarrollo un modelo que usa la red VGG-16[2] como base, debido al desempeño se decidió desarrollar una segunda iteración, en esta se optó por implementar una red similar a la propuesta por Tsung-Yu Lin *et*

al.[3] en la cual proponen una red neuronal convolucional bilineal en donde una imagen es pasado por dos redes neuronales, una red A y una red B, para obtener los descriptores de la imágenes, estos descriptores son de tamaño C*M y M*N respectivamente y son multiplicados usando un producto exterior, el vector resultante se remoleda a un tamaño MN*1 y se le aplica una raíz cuadrada seguido por una normalización l2, finalmente el vector es usado por una capa de clasificación para obtener los resultados.

3.1 Red neuronal usando VGG-16 (VGG)

Este fue el modelo desarrollado en la primera iteración, usa una implementación de VGG-16[2] pre-entrenada en el dataset Imagenet y fue truncada en las capas convolucionales, se le añadió una capa de MaxPooling, una capa de Flatten, una capa densa de 256 neuronas con activación ReLu y una capa densa de 69 neuronas con activación Softmax. Se uso la función de perdida *sparse categorical crossentropy*, el optimizador Adam con una Tasa se aprendizaje de 0.005 y se entreno a lo largo de 50 épocas.

3.1 Red neuronal bilineal usando Xception (B-Xcep)

Este fue el modelo desarrollado durante la segunda iteración, se optó por usar tanto en la red A como en la red B una implementación de la red Xception[4] pre-entrenada en el dataset Imagenet, esta red fue truncada en las capas convolucionales el resultado de las capas convolucionales fue multiplicado usando producto exterior y se obtuvo el vector bilineal, se usó una capa *Flatten* y una capa densa de tamaño 69 con una función de activación softmax para obtener los resultados. Se uso la función de perdida *sparse categorical crossentropy*, el optimizador Adam con una Tasa se aprendizaje de 0.03 y se entrenó a lo largo de 70 épocas.

3.1 Red neuronal bilineal usando VGG-16 (B-VGG)

Este fue el modelo desarrollado durante la segunda iteración, se optó por usar tanto en la red A como en la red B una implementación de la red VGG-16[2] pre-entrenada en el dataset Imagenet, esta red fue truncada en las capas convolucionales el resultado de las capas convolucionales fue multiplicado usando producto exterior y se obtuvo el vector bilineal, se usó una capa *Flatten* y una capa densa de tamaño 69 con una función de activación softmax para obtener los resultados. Se uso la función de perdida *sparse categorical*

crossentropy, el optimizador Adam con una Tasa se aprendizaje de 0.03 y se entrenó a lo largo de 70 épocas.

3.1 Red neuronal bilineal usando InceptionV3 (B-Inc)

Este fue el modelo desarrollado durante la segunda iteración, se optó por usar tanto en la red A como en la red B una implementación de la red InceptionV3 [5] pre-entrenada en el dataset Imagenet, esta red fue truncada en las capas convolucionales el resultado de las capas convolucionales fue multiplicado usando producto exterior y se obtuvo el vector bilineal, se usó una capa *Flatten* y una capa densa de tamaño 69 con una función de activación softmax para obtener los resultados. Se uso la función de perdida *sparse categorical crossentropy*, el optimizador Adam con una Tasa se aprendizaje de 0.03 y se entrenó a lo largo de 70 épocas.

3.1 Red neuronal bilineal usando la implementación propuesta por Tsung-Yu Lin *et al.* (B-CNN)

Este fue el modelo desarrollado durante la segunda iteración, en esta implementación se siguió la propuesta por Tsung-Yu Lin et al. En esta implementación la red A utilizada fue la red D propuesta por K. Simonyan y A. Zisserman[2], esta red utiliza 13 capas convolucionales y 4 capas de MaxPooling, y la red B utilizada fue la red M propuesta por K. Chatfield et al.[6] utiliza 5 capas convolucionales y 2 capas de MaxPooling. El resultado de las capas convolucionales fue multiplicado usando producto exterior y se obtuvo el vector bilineal, se usó una capa *Flatten* y una capa densa de tamaño 69 con una función de activación softmax para obtener los resultados. Se uso la función de perdida *sparse categorical crossentropy*, el optimizador Adam con una Tasa se aprendizaje de 0.03 y se entrenó a lo largo de 70 épocas.

4. Resultados

Se reportan los resultados obtenidos en la sección de validación del dataset, el numero de clases se redujo de 200 a 69 especies, se evaluó con respecto a 3 métricas, estas son exactitud, precisión y sensibilidad.

Los resultados se muestran en la siguiente tabla:

	Exactitud	**Precisión**	**Sensibilidad**

VGG	0.1802	0.2005	0.1763
B-Xcep	0.7308	0.7805	0.7248
B-VGG	0.3481	0.4449	0.3560
B-Inc	0.7185	0.7629	0.7151
B-CNN	0.0543	0.0319	0.0625

Tabla 1. Resultados de los diferentes modelos

Como se puede observar en la tabla el modelo que obtuvo el mejor resultado fue la red bilineal que utiliza las redes Xception, la cual tiene los mejores resultados en las 3 métricas usadas, aunque no se puede usar una comparación con la red propuesta originalmente en el artículo publicado por Tsung-Yu Lin *et al.* debido a que se redujeron el numero de clases, el modelo que usa redes Xception tiene un 73% en comparación con el 84% del mejor modelo reportado por T. Lin.

5. Conclusión

En conclusión, el modelo desarrollado tiene una precisión menor que el propuesto por el artículo publicado por Tsung-Yu Lin *et al.* esto es consecuencia de que no se siguió a pie de la letra la propuesta ya que hay diferencia en el número de épocas y la tasa de aprendizaje, entre otros factores que es posible que haya omitido durante el desarrollo de los modelos. Finalmente, el modelo creado puede ser mejorado en diversas maneras ya sea intentando con *fine-tunning* las redes A y B usadas o usando una combinación de redes diferentes, entre otras posibilidades.

Referencias:

1. Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2022). CUB-200-2011 (1.0) [Data set]. CaltechDATA. https://doi.org/10.22002/D1.20098
2. Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations (ICLR 2015), 1–14.https://arxiv.org/abs/1504.07889
3. T. -Y. Lin, A. RoyChowdhury and S. Maji, "Bilinear CNN Models for Fine-Grained Visual Recognition," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp. 1449-1457, doi: 10.1109/ICCV.2015.170.
4. F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017 pp. 1800-1807. doi: 10.1109/CVPR.2017.195

5. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016 pp. 2818-2826.
6. K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. Ihttps://arxiv.org/abs/1405.3531